

A Study on the Impact of Locality in the Decoding of Binary Cyclic Codes

M. Nikhil Krishnan^a, Bhagyashree Puranik^b, P. Vijay Kumar^c, Itzhak Tamo^d, and Alexander Barg^e

Abstract

In this paper, we study the impact of locality on the decoding of binary cyclic codes under two approaches, namely ordered statistics decoding (OSD) and trellis decoding. Given a binary cyclic code having locality or availability, we suitably modify the OSD to obtain gains in terms of the Signal-To-Noise ratio, for a given reliability and essentially the same level of decoder complexity. With regard to trellis decoding, we show that careful introduction of locality results in the creation of cyclic subcodes having lower maximum state complexity. We also present a simple upper-bounding technique on the state complexity profile, based on the zeros of the code. Finally, it is shown how the decoding speed can be significantly increased in the presence of locality, in the moderate-to-high SNR regime, by making use of a quick-look decoder that often returns the ML codeword.

I. INTRODUCTION

Efficient decoding of general linear block codes has historically been a well-studied problem. The decoding procedure that optimizes the performance of the code is maximum likelihood (ML) decoding. Even though optimal, ML decoding schemes in general become computationally complex as the code-length increases. One of the most developed research directions aimed at reducing the complexity of ML decoding has been trellis decoding and associated trellis representations of codes [1], [2]. Several schemes that approximate ML decoding have been suggested in the literature; among them such reliability-based decoding methods as the Chase algorithm [3], decoding based on ordered statistics [4], and others [5, Ch. 10].

At the same time, significant progress has been achieved in decoding particular code families based on the constraints for the parity check matrix or other structural properties of the codes. Arguably the most far-reaching results along these lines have been accomplished for low-density-parity-check (LDPC) codes and related families based on their iterative decoding [6]. Iterative decoding schemes have also been applied to other codes and a variety of other problems in information and communication theory.

The focus of this paper is a class of codes introduced recently by Gopalan et al. [7] for applications in distributed storage. The main idea of [7] stems from the need to minimize network communication between storage nodes and is formalized in the concept of locality wherein an erased code-symbol can be repaired using a small subset of other symbols of the codeword, usually called the recovery set of the symbol. In the context of linear codes, locality translates into the requirement that every coordinate of the codeword appear in a low-weight parity check equation of the code. Thus, while we do not require that all the parities are of low weight, a linear locally recoverable code (linear LRC code) has a set of low-weight parities whose union contains all the code coordinates. This feature bridges the classes of LRC codes and LDPC codes, and has been used in previous research [8], [9] to prove bounds on the rate of LRC codes with a given minimum distance.

The described connection between LRC and LDPC codes is the starting point of this research which develops in the context of Ordered Statistics Decoding (OSD). Introduced by Fossorier and Lin [4], OSD represents a novel way to decode binary linear block codes based on the reliability of received symbols and provides a flexible trade-off between complexity and performance. Among follow-up papers, the work [10] studies iterative decoding in conjunction with reliability-based decoding, for medium length LDPC codes. This approach is adapted to the decoding of non-sparse codes in the works [11] and [12], where the authors periodically modify parity check matrices to reduce error propagation. The main idea of our work is to decode LRC codes in two stages, beginning with one step of belief propagation decoding using the local parities, and continuing with running OSD decoding on the entire code block. We also pursue another, related line of thought, namely, the impact of locality on trellis decoding (trellis complexity) of linear codes.

Locality constraints introduced in [7] were generalized by Kamath et al. [13] to include stronger local codes and vector alphabets. They were further extended by Sasidharan et al. [14] who defined a hierarchical structure of several levels of locality constraints. At the same time, Wang and Zhang [15] suggested another generalization LRC codes wherein every coordinate

^a Department of Electrical Communication Engineering, Indian Institute of Science, Bangalore, India, Email: nikhilkrishnan.m@gmail.com. Research supported through Visvesvaraya PhD Scheme for Electronics & IT awarded by Department of Electronics and Information Technology, Govt. of India.

^b Department of Electrical Communication Engineering, Indian Institute of Science, Bangalore, India, Email: bhagya.puranik@gmail.com

^c Department of Electrical Communication Engineering, Indian Institute of Science, Bangalore, India, Email: pvk1729@gmail.com. P. V. Kumar is also an Adjunct Research Professor at the University of Southern California. His research is supported in part by NSF under Grant No. 1421848 and in part by the joint UGC-ISF research program.

^d Department of EE-Systems, Tel Aviv University, Tel Aviv, Israel, Email: zactamo@gmail.com. Research is supported by ISF grant no. 1030/15 and the NSF-BSF grant no. 2015814.

^e Dept. of ECE and ISR, University of Maryland, College Park, USA, and Inst. Inform. Trans. Probl. (IITP), Russian Academy of Sciences, Moscow, Russia, Email: abarg@umd.edu. Research is supported in part by NSF grants CCF1422955 and CCF1618603.

has several independent (disjoint) recovery sets, much like orthogonal parities employed much earlier in threshold decoding of linear codes [16]. When we wish to emphasize the fact that our codes possess several recovery sets, we call them LRC codes with availability. In this paper we observe that the decoding algorithm we are having in mind is fitted well to hierarchical LRC codes and LRC codes with availability, and therefore use them in our examples to quantify the improvements in decoding performance.

The main class of codes studied in this paper is cyclic LRC codes. These codes were initially studied by Tamo et al. [17] and are derived from a more general family of Reed-Solomon-like LRC codes defined earlier in [18]. We note that decoding of cyclic codes has been of interest in coding theory due to the rich structure possessed by these codes. Of the large number of works devoted to ML decoding of cyclic codes, we mention the paper by Vardy and Be'ery [19] which targeted algebraic properties of cyclic codes similar to [17], albeit with no connection to locality constraints. The main results in [19] pertain to reducing trellis complexity of cyclic codes by studying the direct-sum and concurring-sum structures existing in BCH codes. The direct-sum structure connects their results to LRC codes with disjoint repair groups, and this is the setting that we pursue in this paper. We note that the constructions in [18] and [17] also form codes with disjoint recovery sets.

Our results: In the first part of this paper, we suggest a non-iterative, locality-aware decoding scheme for binary cyclic codes with disjoint repair groups and hierarchical locality or availability, built upon the classical OSD [4]. Performance gains in terms of SNR are obtained via a single round of belief propagation based on local parity checks alone. For codes with disjoint/hierarchical locality, decoding delays are reduced by making use of a simple, quick-look ML decoding stage that relies on the presence of local codes with disjoint supports.

In the second part, which deals with trellis decoding of cyclic codes having composite code lengths, we discuss how to design cyclic codes having lower computational complexity through careful introduction of locality. Based on a coordinate ordering that aligns direct-sum structures [19], we obtain an upper bound on state complexity profile of a code with or without locality. Further, the frequency domain approach that we adopt, also helps in the better understanding of cyclic codes with (r, δ) -locality and hierarchical locality properties.

II. LOCALITY AND CYCLIC CODES

A. Preliminaries: Codes with Locality

All the codes discussed in this paper are assumed to be linear of length n and dimension k over a field \mathbb{F}_q unless specified otherwise, and we write their parameters as $[n, k]$. Let $[l] \triangleq \{1, 2, \dots, l\}$ and $[0, l] \triangleq \{0, 1, \dots, l\}$.

The codes studied in this paper belong to the class of LRC codes [7]. We begin with a definition, limiting ourselves to the linear case.

Definition II.1. [7] Let $0 \leq i \leq n - 1$. The i^{th} coordinate of \mathcal{C} is said to have *locality* r if, for every codeword $\underline{c} \triangleq (c_0, c_1, \dots, c_{n-1}) \in \mathcal{C}$, there exists a subset $\mathcal{I}_i \subset [0, n - 1] \setminus \{i\}$ with $|\mathcal{I}_i| \leq r$ such that c_i is a linear combination of the symbols $\{c_j, j \in \mathcal{I}_i\}$. i.e., $c_i = \sum_{j \in \mathcal{I}_i} \lambda_{i,j} c_j$, where $\lambda_{i,j} \in \mathbb{F}_q$ and $\lambda_{i,j} \neq 0$. Here, both \mathcal{I}_i and $\{\lambda_{i,j}\}$ depend on i but not on \underline{c} . A code \mathcal{C} is said to have r -all-symbol locality or simply, *locality* r if, all the coordinates $\{i : 0 \leq i \leq n - 1\}$ have locality r . A code having r -all-symbol locality is called as an LRC with locality r .

The subset \mathcal{I}_i is called the *recovery set* of the coordinate i . It is clear that every coordinate $j \in \mathcal{I}_i \cup \{i\}$ can be found from the remaining coordinates in this set, so we call the set $\mathcal{I}_i \cup \{i\}$ a *repair group*. Most constructions of LRC codes in the literature, including those in [18], [17], [20] and the codes considered in this paper, have the property of disjoint repair groups (for brevity we call them codes with disjoint locality).

The following generalization of this definition was suggested in [13].

Definition II.2. [13] For $0 \leq i \leq n - 1$, the i^{th} symbol of a linear code \mathcal{C} over \mathbb{F}_q with generator matrix \mathbf{G} is said to have (r, δ) -code-symbol locality if there exists $\mathcal{I}_i \subseteq [0, n - 1]$ such that $i \in \mathcal{I}_i$ and $|\mathcal{I}_i| \leq r + \delta - 1$, where minimum distance of the punctured code $\mathcal{C}_{\mathcal{I}_i}$ is at least δ . Such a smaller length punctured code will be referred to as a local code. Its corresponding parity checks are termed as local parity checks.

Further, \mathcal{C} is said to possess (r, δ) -information locality if there exists an information set $\mathcal{I} \subseteq [0, n - 1]$ such that $\mathbf{G}_{\mathcal{I}}$ (\mathbf{G} punctured to the coordinates in \mathcal{I}) has rank k and all $i \in \mathcal{I}$ have (r, δ) -code-symbol locality. Similarly, if every code-symbol c_i , $i \in [0, n - 1]$ has (r, δ) -code-symbol locality, the code is said to have (r, δ) -all-symbol locality. Codes with locality r defined above [7] form a particular case of this definition for $\delta = 2$. For any code $r \leq k$, because (unless the distance of \mathcal{C} is 1), \mathbf{G} contains k linearly independent columns that do not include any chosen coordinate i .

Remark II.1. We note that the connection between LRC codes and LDPC codes extends to codes having (r, δ) -all-symbol locality. Specifically, for $\delta > 2$ the sets of local parities play the same role as local codes in Generalized LDPC codes [21], [22]. This analogy is useful in the derivation of lower bounds on the rate of (r, δ) LRC codes; see [23].

Definition II.3. [15] A code with r -locality is said to have t -availability if for every $i \in [0, n - 1]$, there exist t distinct codewords $\underline{c}_1, \underline{c}_2, \dots, \underline{c}_t$ in the dual code \mathcal{C}^\perp such that:

- (i) for any $a \in [t]$, $i \in \text{supp}(\underline{c}_a)$,
- (ii) for any $a \in [t]$, $|\text{supp}(\underline{c}_a)| \leq r + 1$,
- (iii) for any distinct indices $a, b \in [t]$, $\{\text{supp}(\underline{c}_a)\} \cap \{\text{supp}(\underline{c}_b)\} = \{i\}$,

where the support of a codeword $\underline{c} \in \mathcal{C}$ is defined as $\text{supp}(\underline{c}) \triangleq \{i : c_i \neq 0\}$.

We define the support of a set of codewords \mathcal{D} as $\text{supp}(\mathcal{D}) \triangleq \cup_{\underline{c} \in \mathcal{D}} \text{supp}(\underline{c})$.

Definition II.4. [14] A code is said to possess *two-level hierarchical locality* with locality parameters $[(r_{\text{mid}}, \delta_{\text{mid}}), (r_{\text{base}}, \delta_{\text{base}})]$, if

- (i) for every i^{th} symbol, there exists a ‘middle’ local code, $\mathcal{C}_i^{\text{mid}}$ with $i \in \text{supp}(\mathcal{C}_i^{\text{mid}})$ such that $\dim(\mathcal{C}_i^{\text{mid}}) \leq r_{\text{mid}}$ and $d_{\min}(\mathcal{C}_i^{\text{mid}}) \geq \delta_{\text{mid}}$;
- (ii) furthermore, for each j^{th} symbol of the code $\mathcal{C}_i^{\text{mid}}$, there exists a ‘base’ local code $\mathcal{C}_{i,j}^{\text{base}}$ with $j \in \text{supp}(\mathcal{C}_{i,j}^{\text{base}}) \subseteq \text{supp}(\mathcal{C}_i^{\text{mid}})$ such that $\dim(\mathcal{C}_{i,j}^{\text{base}}) \leq r_{\text{base}}$ and $d_{\min}(\mathcal{C}_{i,j}^{\text{base}}) \geq \delta_{\text{base}}$.

This definition can be extended to more than two levels of local code hierarchy.

B. Cyclic Punctured Codes and Shortened Codes

Consider an $[n, k]$ cyclic code \mathcal{C} over a field \mathbb{F}_q such that $(n, q) = 1$ and $n | (q^m - 1)$, where m is the multiplicative order of q modulo n . Let $\alpha \in \mathbb{F}_{q^m}$ be a primitive n^{th} root of unity. Given a codeword $\underline{c} = (c_0, c_1, \dots, c_{n-1})$, let $c(x) = \sum_{i=0}^{n-1} c_i x^i$. Let $\alpha^{i_1}, \alpha^{i_2}, \dots$ be the set of common zeros of these polynomials in the set of n^{th} roots of unity. By abuse of terminology, we call the set $\mathcal{S}(\mathcal{C}) \triangleq \{i_1, i_2, \dots\}$ the set of the zeros of the code \mathcal{C} . Recall that if i is a zero of \mathcal{C} , then so is $qi \bmod n$, and the set of distinct elements among $\{(q^j i) \bmod n, j = 1, 2, \dots\}$ is called the cyclotomic coset of i . We note that a cyclic code is fully described by specifying only the representatives of cyclotomic cosets.

Let $\mathcal{I} \subseteq [0, n-1]$ be a subset of code coordinates. The codes having length $|\mathcal{I}|$ obtained by shortening and puncturing \mathcal{C} to \mathcal{I} are denoted by $\mathcal{C}^{\mathcal{I}}$ and $\mathcal{C}_{\mathcal{I}}$, respectively. Let $n_1 | n$ and $\nu \triangleq \frac{n}{n_1}$. We define the following set: $\mathcal{I}_i \triangleq \{i, i + \nu, i + 2\nu, \dots, i + n - \nu\}$, where $0 \leq i \leq \nu - 1$, which may be regarded as being obtained by shifting and then decimating the set $[0, n-1]$. We will call each such set a *length- n_1 support set*. It can be verified that $\mathcal{C}_{\mathcal{I}_i}$ ’s (and similarly, $\mathcal{C}^{\mathcal{I}_i}$ ’s) are all identical for $0 \leq i \leq \nu - 1$ and are moreover cyclic codes of length n_1 . We refer to these codes as length- n_1 punctured (and similarly, shortened) codes of \mathcal{C} . The zeros of these codes will be assumed to be computed with respect to the exponents of $\beta = \alpha^\nu$. We cite below a result from [19] which relates the zeros $\mathcal{S}(\mathcal{C}^{\mathcal{I}_i})$ of the shortened code to $\mathcal{S}(\mathcal{C})$:

Lemma II.1. [19] *The zeros of the shortened code $\mathcal{C}^{\mathcal{I}_i}$ and the zeros of the code \mathcal{C} are related as follows:*

$$\mathcal{S}(\mathcal{C}^{\mathcal{I}_i}) = \left\{ \lambda \bmod n_1 : \lambda \in \mathcal{S}(\mathcal{C}) \right\}.$$

We derive the following analogous lemma for punctured codes.

Lemma II.2. *The zeros of the punctured code $\mathcal{C}_{\mathcal{I}_i}$ and the zeros of the code \mathcal{C} are related as follows:*

$$\mathcal{S}(\mathcal{C}_{\mathcal{I}_i}) = \left\{ \lambda \bmod n_1 : \{\lambda, \lambda + n_1, \dots, \lambda + (\nu - 1)n_1\} \subseteq \mathcal{S}(\mathcal{C}) \right\}.$$

The proof of this lemma is given in Appendix A. We call the set $\mathcal{T}_a^{(n_1)} \triangleq \{a, a + n_1, \dots, a + (\nu - 1)n_1\}$, where $a \in [0, n_1 - 1]$, a *locality train* corresponding to n_1 .

Remark II.2. If $|\mathcal{S}(\mathcal{C}_{\mathcal{I}_i})| > 0$, then the code \mathcal{C} has r -all-symbol locality, with $r = n_1 - |\mathcal{S}(\mathcal{C}_{\mathcal{I}_i})|$.

C. Zeros and Locality

A sufficient condition for a cyclic code to have r -all-symbol locality in terms of its zeros was proved in [17] and is stated in the theorem that follows. We use Lemma-II.2 to give an alternate proof of this theorem.

Theorem II.3. [17] *Let r be a positive integer such that $(r+1) | n$. If $\{j, j + (r+1), j + 2(r+1), \dots, j + (\frac{n}{r+1} - 1)(r+1)\} \subseteq \mathcal{S}(\mathcal{C})$ for some $j \in [0, r]$, then \mathcal{C} has r -all-symbol locality.*

Proof. Let $n_1 = r + 1$ and hence, $\nu = \frac{n}{r+1}$. From Lemma-II.2, it follows that $j \in \mathcal{S}(\mathcal{C}_{\mathcal{I}_i})$ for all $0 \leq i \leq \nu - 1$. Therefore, any nonzero codeword (polynomial) in each length- $(r + 1)$ punctured code $\mathcal{C}_{\mathcal{I}_i}$ has at least two nonzero coefficients. In other words, the distance $d_{\min}(\mathcal{C}_{\mathcal{I}_i}) \geq 2$, which implies our claim. \square

The above result can be extended to the case of (r, δ) -locality for $\delta > 2$, and is given below. This theorem also generalizes a similar result that appears in [20].

Theorem II.4. Let n_1 be a positive integer such that $n_1|n$ and $\mathcal{X} \subseteq [0, n_1 - 1]$. Let δ be the guaranteed minimum distance for the length- n_1 cyclic code having \mathcal{X} and its cyclotomic cosets as zeros. If $\cup_{j \in \mathcal{X}} \{j, j + n_1, j + 2n_1, \dots, j + n - n_1\} \subseteq \mathcal{S}(\mathcal{C})$, then \mathcal{C} has (r, δ) -all-symbol locality with $r = n_1 - \delta + 1$.

Corollary II.5. [20] Let $n_1|n$, $\mathcal{X} = \{i_1, i_2, \dots, i_{\delta-1}\}$ and d be an integer satisfying $(n_1, d) = 1$, where $0 \leq i_1 \leq i_2 \leq \dots \leq i_{\delta-1} \leq n_1 - 1$, $i_l = i_1 + d(l - 1)$, $1 \leq l \leq \delta - 1$. If $\cup_{j \in \mathcal{X}} \{j, j + n_1, j + 2n_1, \dots, j + n - n_1\} \subseteq \mathcal{S}(\mathcal{C})$, then \mathcal{C} has (r, δ) -all-symbol locality with $r = n_1 - \delta + 1$.

Proof. Follows from application of the BCH bound. □

We invoke Lemma-II.2 to show the existence of hierarchical locality in the example below.

Example II.1. Consider the $[63, 33]$ binary cyclic code \mathcal{C} having zeros $\{0, 1, 3, 5, 7, 21, 27\}$ along with the elements of their respective 2-cyclotomic cosets. On account of Lemma II.2, there are three local codes (punctured codes) of length 21 and dimension 15 whose zeros are $\{0, 3, 6, 7, 12, 14\}$ modulo $n_1 = 21$. Furthermore, each length-21 punctured code is a disjoint union of three length-7 punctured codes, each of which has $\{0\}$ as its set of zeros. From this, it follows that the hierarchical locality parameters of the code are given by $r_{\text{mid}} = 15$, $\delta_{\text{mid}} \geq 3$, $r_{\text{base}} = 6$, $\delta_{\text{base}} = 2$.

III. A LOCALITY-AWARE MODIFICATION OF THE ORDERED STATISTICS DECODING (OSD) ALGORITHM

A. Locality-Aware Modification of the OSD Algorithm

We summarize the OSD algorithm due to Fossorier et al. [4] as follows. We consider the channel with additive white Gaussian noise (AWGN), where the transmitted vector is $\underline{x} = (x_0 \ x_1 \ \dots \ x_{n-1})$, with $x_i \in \{+1, -1\}$. The received vector is $\underline{y} = \underline{x} + \underline{n}$, with $n_i \sim \mathcal{N}(0, \frac{N_0}{2})$, and N_0 is the power spectral density of the noise. Under the OSD algorithm, the received vector is first ordered in decreasing order of the $|y_i|$'s, which are regarded as reliability values. We call the first k bits in this order which are linearly independent as Most Reliable Independent (MRI) symbols. We clip their values to the nearest value of $\{1, -1\}$ and regard them as k information bits of the codeword. The corresponding codeword in \mathcal{C} , is selected as the order-0 OSD algorithm decoded codeword. This may be followed by an order- l reprocessing stage, where, for $0 \leq i \leq l$, all possibilities of i bits from the k MRI bits will be flipped to obtain $M = \sum_{i=0}^l \binom{k}{i}$ information sets. The M information sets are mapped to their respective codewords and the codeword with least Euclidean distance from \underline{y} will be declared as the order- l OSD output.

In this section, we primarily consider binary linear codes having either (a) disjoint or (b) hierarchical locality. In the case of disjoint locality, the code is assumed to have r -all-symbol-locality, with disjoint supports for local (single parity check) codes. For codes with h -level hierarchical locality (say, $h = 2$) local codes within each level are assumed to have disjoint supports. Furthermore, the base code parity checks here are assumed to include single-parity checks.

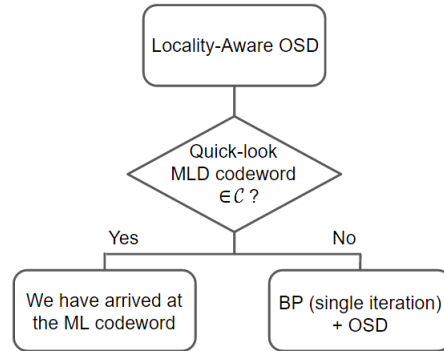


Fig. 1: An illustration of the Locality-Aware OSD algorithm.

In the locality-aware OSD algorithm, we first consider a supercode of \mathcal{C} denoted by \mathcal{C}_{loc} , whose dual code is precisely the space spanned by the local parity checks on \mathcal{C} . The \mathcal{C}_{loc} is a concatenation (direct sum) of single parity-check codes (similarly, middle codes) for the disjoint (similarly, hierarchical) locality scenario. We carry out ML decoding (which we call quick-look ML decoding, Q-MLD) on the received vector, \underline{y} with the underlying code assumed to be \mathcal{C}_{loc} . The disjoint supports of the local codes permit the local codes to be ML-decoded independently.

1) *Disjoint recovery sets:* In this case, ML decoding of a local code can be carried out via a simple, 2-step procedure: (i) hard-decision decoding of the individual symbols (ii) if this does not yield a valid codeword in the single parity-check code, flip the least reliable bit to obtain the ML local codeword. A little thought will show that, if the n -length output vector obtained by putting together the individually ML-decoded local codewords results in a codeword in \mathcal{C} , the decoding procedure can be halted and we have the ML codeword in \mathcal{C} . This simple, Q-MLD trick works really well with medium-length codes,

particularly in the moderate-to-high SNR range. Suppose for simplicity, that there are $\frac{n}{n_l}$ single-parity locality checks of equal length n_l . It is straightforward to derive the following lower bound on the probability that Q-MLD will succeed:

$$P_{\text{success}} \geq \left(1 - \binom{n_l}{2} \left[1 - Q\left(-\sqrt{\frac{4}{N_0}}\right)\right]\right)^{\frac{n}{n_l}} \quad (1)$$

For a length-63 code having nine length-7 single parity checks for locality, $P_{\text{success}} \approx 0.82$ (empirical) at SNR of 3.5dB while the estimate given by (1), equals 0.77. At 4.5dB, the corresponding numbers are given by 0.94 and 0.93, respectively. Thus at 4.5dB SNR, $> 93\%$ of the time, the stage-1 decoding will be stopped, after arriving at the optimal codeword choice. This helps in reducing average complexity and decoding delay, when used in conjunction with a buffer. If however, the put-together n -length vector is not a codeword in \mathcal{C} , a single round of belief propagation based on the single-parity checks alone is carried out, followed by OS order- l decoding.

2) *Hierarchical locality case*: In the hierarchical case, we arrive at the ML codeword for \mathcal{C}_{loc} by carrying out some decoding procedure, for instance trellis decoding, of each middle code. It is possible to design high-rate codes with hierarchical locality, where the middle codes have low trellis-complexity. If Q-MLD does not return a codeword in \mathcal{C} , single parity checks belonging to base local codes will be effectively utilized by the alternate approach having belief propagation and OSD. As middle codes can be designed to be strong codes, this will be a lower probability event compared to the previous case of Q-MLD based on single parity checks.

To illustrate this reasoning, we studied the performance of the code given in Example II.1. Consider the pair of embedded codes $\mathcal{C}_2 \subset \mathcal{C}_1$, where \mathcal{C}_1 has zeros $\{0, 1, 3, 5, 7, 21\}$ (and their cyclotomic cosets) and the code \mathcal{C}_2 in addition to these has zeros $\{27, 45, 54\}$. Here, \mathcal{C}_2 is the code with hierarchical locality given in Example II.1. The code \mathcal{C}_1 has disjoint locality with $r = 6$ owing to the presence of zeros at $0, 7, 14, \dots, 56$. We simulated the success rate of Q-MLD for \mathcal{C}_1 at SNR 5dB, relying on disjoint repair groups of length 7 and their respective parity equations, and found it to be ≈ 0.9650 . We can further improve this outcome by performing trellis decoding of the middle codes of the code \mathcal{C}_2 , then the probability of overall decoding success rises to approximately 0.9994. The trellis decoding steps using the $([0, \infty], +, \min)$ semiring (see [24, E.g. 2.5]) add 587 real calculations (412 additions and 175 comparisons) for each of the three length-21 middle codes. Thus, we have a total of 1761 real calculations (1236 additions and 525 comparisons).

If Q-MLD fails, we perform a simple, one round iteration of BP on a bipartite graph constructed using the single parity local checks having disjoint supports. For each received coordinate y_i , we compute the corresponding log-likelihood ratio (LLR) $\ell_i^{(0)}$ ($= \frac{4y_i}{N_0}$, in this case) and feed it to the BP decoder. Let $\{i_1, i_2, \dots, i_L\}$ be the coordinates belonging to a local parity check. Upon completing one round of the BP decoding, we form an updated vector of LLRs, $\underline{\ell}^{(1)} \triangleq (\ell_1^{(1)}, \ell_2^{(1)}, \dots, \ell_n^{(1)})$ obtained using the following rule [6]:

$$\ell_{i_j}^{(1)} = 2 \tanh^{-1} \left[\prod_{a=1, a \neq j}^L \tanh \left(\frac{\ell_{i_a}^{(0)}}{2} \right) \right] + \ell_{i_j}^{(0)} \quad (2)$$

The vector $\underline{\ell}^{(1)}$ is then submitted to the OSD decoder of order 0 (or higher).

Performing one round of message passing (BP) before OSD is achieved at a small computation cost of $O(n)$ real multiplications and additions along with $O(n)$ \tanh and \tanh^{-1} computations. The conventional OSD order-0 algorithm requires $O(n \log n)$ real comparisons and $O(nk^2)$ binary additions/multiplications. For OSD of order $l \geq 1$, the total cost is $O(nk^l)$ real and $O(nk^{l+1})$ binary additions/multiplications. By certain sufficiency tests given in [4], complexity of OSD can be further reduced.

If the code has t -availability, for each code-symbol, we will update the log-likelihood ratios based on an iteration of BP on the t associated orthogonal parity checks (APP decoding, [16]). This will incur $O(nt)$ \tanh , \tanh^{-1} computations and $O(nrt)$ real multiplications.

B. Performance Evaluation Through Simulations

In the following, we present simulation results which suggest that locality amongst code symbols can be effectively used to obtain significant gains in SNR with negligible complexity overhead in the BP+OS stage. Note that these performance gains are in addition to the reductions in average complexity and delays obtained through the use of Q-MLD. We reiterate that the BP+OS scheme that we use is non-iterative and non-adaptive in nature, making it easier to implement, and yet it gives significant performance gains (see [11], [12] for a discussion of adaptive and iterative decoding techniques).

Without loss of generality, we assume transmission of $\underline{1}$ (the all-zero codeword) in the simulations. Fig. 2 presents results relating to two codes; (i) \mathcal{C} : the [255, 206] BCH code having defining zeros at $\{0, 1, 3, 5, 7, 9, 11\}$ and their 2-cyclotomic cosets (ii) \mathcal{B} : a [255, 192] code with locality, $r = 16$, obtained from \mathcal{C} using Theorem II.3. The code \mathcal{B} has defining zeros at $\{0, 1, 3, 5, 7, 9, 11, 17, 51, 85, 119\}$ and their 2-cyclotomic cosets. To bring out the impact of introducing locality and taking advantage of it, conventional OSD of order-0, 1, and 2 is carried out for both codes and in addition, locality-aware OSD with order-0, 1 and 2 is carried out for \mathcal{B} .

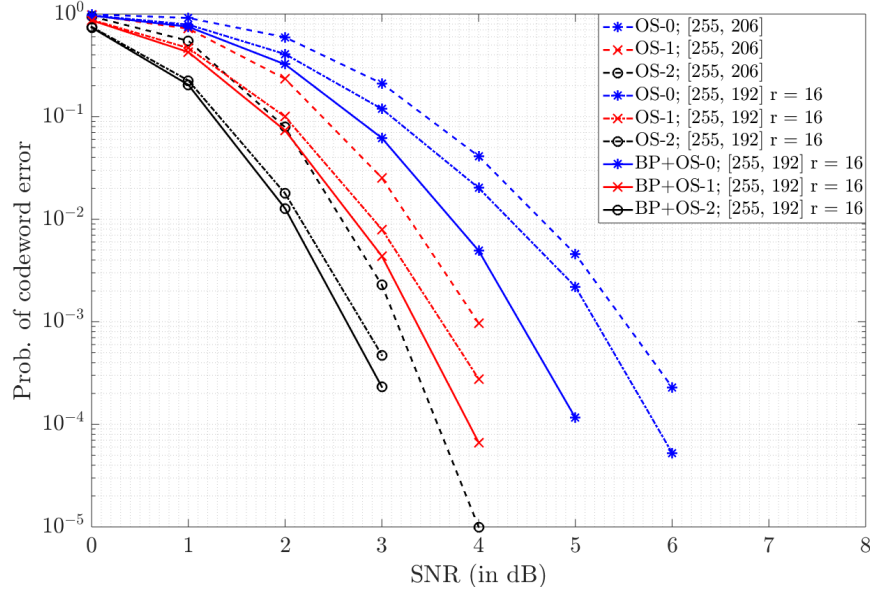


Fig. 2: Comparison of performance of a $[255, 206]$ code without locality and a $[255, 192]$ code with locality parameter $r = 16$

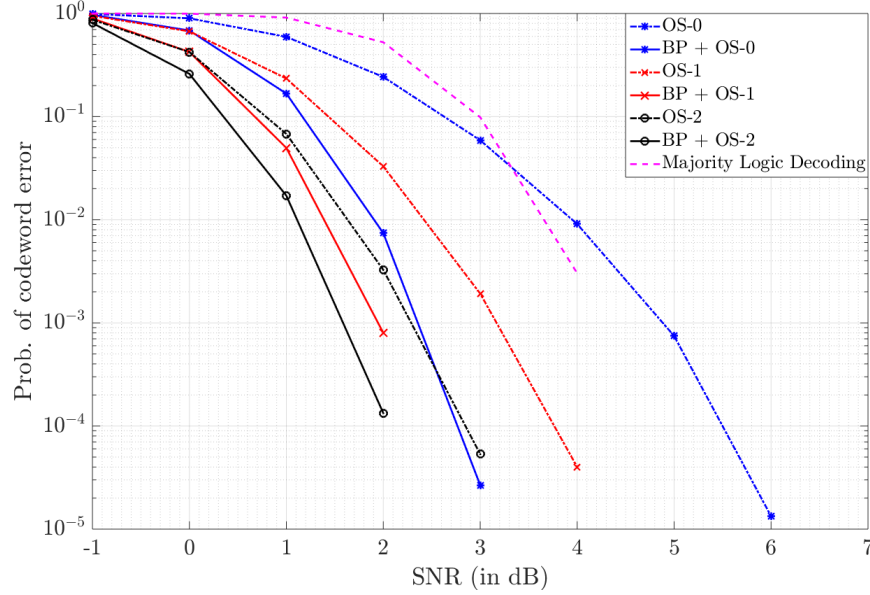


Fig. 3: Comparison of OSD, OSD+BP, and majority logic decoding of the type-I DTI code with parameters $[255, 175]$, $r = 15$, $t = 16$.

In Fig. 3, we consider the $[255, 175]$ majority logic decodable code [5, p.290]. This code has a doubly transitive automorphism group and inherently possesses availability, so we obtain $t = 16$ with $r = 15$. We perform majority logic codeword decoding, conventional OS of order-0, 1 or 2. In the locality-aware OSD scheme, likelihoods will be obtained for each symbol, based on t orthogonal parity checks, followed by OSD of order-0, 1 and 2. Here, we observe that BP+order-0 OSD outperforms even conventional order-2 OSD in the higher SNR range. For both the simulations, similar trends have been observed with respect to Bit-Error-Rate (BER) performance as well, in the same SNR range. For codes having overlapping local parity checks, preliminary experiments are inconclusive and call for further study.

IV. LOCALITY IN TRELLIS DECODING

In this section, we change the subject and instead of pursuing improved performance of codes in terms of the error probability, study the benefits of having locality for reducing *decoding complexity*, focusing on trellis decoding. Trellis representation of a linear block code along with Viterbi decoding enables one to perform efficient soft-decision ML decoding [2], [25], [5, Ch. 14].

Let $\underline{c} = (c_0, c_1, \dots, c_{n-1})$ be a codeword of \mathcal{C} and $0 \leq i \leq n$. The past shortened code at level- i , $\mathcal{P}_{(i)}$ is defined to be the subcode of \mathcal{C} shortened to the first i coordinates, i.e., coordinates $[0, i-1]$. Thus, we have $\mathcal{P}_{(i)} = \{\underline{c} \in \mathcal{C} : c_i = c_{i+1} = \dots = c_{n-1} = 0\}$. The future shortened code of \mathcal{C} at level- i , $\mathcal{F}_{(i)}$, is the subcode of \mathcal{C} shortened to the last $(n-i)$ coordinates $[i, n-1]$ i.e., $\mathcal{F}_{(i)} = \{\underline{c} \in \mathcal{C} : c_0 = c_1 = \dots = c_{i-1} = 0\}$. Similarly, $\mathcal{P}^{(i)}$ and $\mathcal{F}^{(i)}$ are the codes obtained by puncturing \mathcal{C} to the first i and last $(n-i)$ coordinates, respectively. Therefore, we have $\mathcal{P}^{(i)} = \{(c_0, c_1, \dots, c_{i-1}) : \underline{c} \in \mathcal{C}\}$ and $\mathcal{F}^{(i)} = \{(c_i, c_{i+1}, \dots, c_{n-1}) : \underline{c} \in \mathcal{C}\}$. The corresponding dimensions of these four codes at level- i are denoted by p_i , f_i , p^i and f^i . By definition we assume that $p_0 = p^0 = f_n = f^n = 0$.

The trellis diagram is a finite directed graph with a set of vertices and edges (V, E) constructed from the parity-check matrix of the code [2], [5, Ch.9]. We denote the set of vertices at level i of the trellis by V_i . The *state complexity* at level i is defined as $s_i \triangleq \log_q |V_i|$, $i \in [0, n]$. The following relations hold [26]:

$$s_i = k - p_i - f_i = p^i - p_i = p^i + f^i - k = f^i - f_i \quad (3)$$

$$s_i \leq \min\{k, n - k\} \quad (4)$$

The computational complexity of the Viterbi algorithm using trellis decoding is shown to be $2|E| - |V| + 1$ [24], where $|E|$ and $|V|$ are the total number of edges and vertices in the trellis, respectively.

Here, we consider cyclic codes having composite code lengths, with $n|(q^m - 1)$. The code need not have locality. We assume the natural coordinate ordering that arises from the direct-sum structure [19] existing in these codes. Let \mathcal{F}_n be the set of all factors of n except 1 and n . We will call a subset of factors $\tau \triangleq \{x_1, x_2, \dots, x_L\} \subseteq \mathcal{F}_n$ a *chain* if $x_a | x_b$ and $x_a \neq x_b$ for all $a < b$.

In the remainder of this section, we introduce a coordinate permutation which we will use in our estimates of trellis complexity. We start with a chain τ . Using the terminology introduced in Sec. II-B, for each $l \in [L]$ there exist $\frac{n}{x_l}$ length- x_l support sets. Each of these sets is of the form $\{a, a+x_l, \dots, a+(\frac{n}{x_l}-1)x_l\}$, for some $a \in [0, x_l-1]$. As explained in Sec. II-B, each of these subsets supports a shortened cyclic code and a punctured cyclic code. Given a chain τ and the associated support sets, we permute the code's coordinates so that each support set forms a block of contiguous coordinates. We will denote the resulting ordering of the set $[0, n-1]$ by γ . Under this ordering, each length- x_l support set occupies the coordinates $[(j-1)x_l, jx_l-1]$ for some $j \in \{1, \dots, n/x_l\}$. We note that the code obtained by permuting the coordinates of \mathcal{C} is not necessarily cyclic.

The coordinate ordering γ can be defined with respect to a tree of multiplicative subgroups of the group $\mathbb{F}_{q^m}^*$ generated by each of the following field elements: $\alpha^{\frac{n}{x_1}}, \alpha^{\frac{n}{x_2}}, \dots, \alpha^{\frac{n}{x_L}}$ and their cosets (here α is a primitive element of \mathbb{F}_{q^m}). We use the elements of the field as locators of the code's coordinates, labeling $i \in [0, n-1]$ as α^i . The easiest way to explain the ordering γ is by example. Consider a cyclic code of length $n = 16$ over the field \mathbb{F}_{17} and consider the chain $\tau = \{4, 8\}$. Let α be a primitive element of \mathbb{F}_{17} . Further, let $H_1 = \mathbb{F}_{17}^* = \langle \alpha \rangle$, $H_2 = \langle \alpha^2 \rangle$, and $H_3 = \langle \alpha^4 \rangle$ be the entire multiplicative group of the field and its subgroups associated with the chain τ , where $\langle \eta \rangle$ denotes the multiplicative group generated by η . In Fig. 4 we show a tree of subgroups and their cosets. The leaves of this tree taken in the natural order (left-to-right according to the figure) define the coordinate ordering γ which is explicitly given as $\gamma = (0, 4, 8, 12, 2, 6, 10, 14, 1, 5, 9, 13, 3, 7, 11, 15)$.

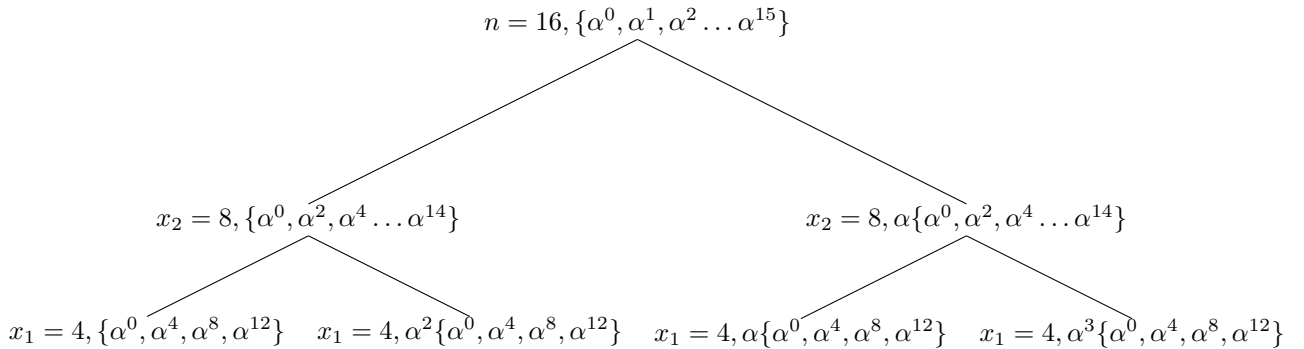


Fig. 4: For a length-16 cyclic code over \mathbb{F}_{17} and $\tau = \{4, 8\}$, the ordering $\gamma = (0, 4, 8, 12, 2, \dots, 7, 11, 15)$.

To give another example, let $n = 63$, $\tau = \{7, 21\}$, $\beta = \alpha^3$, $\gamma = \beta^3$, $H_1 = \langle \alpha \rangle$, and $H_2 = \langle \beta \rangle$, $H_3 = \langle \gamma \rangle$. The corresponding ordering γ is illustrated in Fig. 5.

If the code \mathcal{C} has the locality property, then using the results in Section II-C, we observe that this ordering essentially brings the disjoint repair groups to x_l contiguous positions. In the following, by abuse of notation we do not differentiate between the code \mathcal{C} and its permuted (equivalent) version.

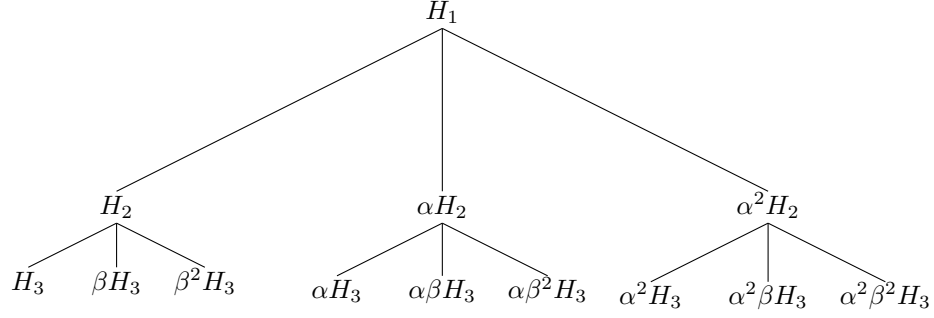


Fig. 5: Case $n = 63$, $\tau = \{7, 21\}$. Each leaf gives 7 coordinates sorted by increasing powers of α . For instance, let $\beta H_3 = \{\alpha^{i_1}, \alpha^{i_2}, \dots, \alpha^{i_7}\}$, with $0 \leq i_1 < i_2 < \dots < i_7 < n$. In this case, the leaf βH_3 gives the (ordered) coordinates i_1, i_2, \dots, i_7 . The γ -ordering corresponds to the natural order of the leaves from left to right.

A. Using locality to reduce state complexity of cyclic codes

Given an $[n, k]$ cyclic code \mathcal{C} over \mathbb{F}_q , it is possible to employ locality to construct a subcode $\mathcal{C}_{\text{new}} \subset \mathcal{C}$ with low trellis complexity. Apart from the obvious dependency of state complexity on the dimension of the punctured code dimensions (viz. (3)), the approach of this section is further motivated by empirical analyses suggesting locality to be a ubiquitous feature in cyclic codes of composite length that have low trellis complexity, with respect to the chain-based orderings we consider. For instance, we considered (via a computer search) all binary cyclic codes of length $n = 63$ having low trellis complexity (edge count ≤ 50000 , after fixing the chain to be $\tau = \{7, 21\}$). Further, we restricted the search to moderate code-rates, $0.2 \leq \frac{k}{n} \leq 0.8$. This was because, at very high and very low rates, complexity is bound to be low due to (4). Interestingly, the computer search revealed that all the codes being considered have r -all-symbol locality, with $r \leq 19$ (the codes of dimension $k \leq 19$ trivially satisfy this condition), and on an average, the codes had a locality of 10.

Let $n_j | n$. From Lemma-II.2, if the set of zeros of the code \mathcal{C} contains a locality train $\mathcal{T}_b^{(n_j)} = \{b, b+n_j, b+2n_j, \dots, b+n-n_j\}$ for some $b \in [0, n_j - 1]$, then \mathcal{C} has the locality property, with length- n_j punctured codes as local codes. Our idea in the remainder of this section is to show that adding locality trains to the set of zeros of the code can be used to reduce the state complexity of resulting subcode. Let $b \in [0, n_j - 1]$ and let $\mathbb{C}_b^{n_j}$ be the q -cyclotomic coset modulo n_j that contains b . Adding $\mathcal{T}_b^{(n_j)}$ to $\mathcal{S}(\mathcal{C})$ amounts to taking the subcode $\mathcal{C}_{\text{new}} \subset \mathcal{C}$ whose set of zeros is

$$\mathcal{S}(\mathcal{C}_{\text{new}}) = \bigcup_{h \in \mathbb{C}_b^{n_j}} \mathcal{T}_h^{(n_j)}.$$

Example IV.1. Consider a binary cyclic code \mathcal{C} of length $n = 63$ with zeros $\{1, 2, 4, 8, 16, 32, 3, 6, 12, 24, 48, 33\}$. We add to $\mathcal{S}(\mathcal{C})$ the following locality trains corresponding to 21: $T_1 = \{0, 21, 42\}$, $T_2 = \{3, 24, 45\}$. Note that adding the sets T_1, T_2 is equivalent to adding the zeros $\{0, 21, 45\}$ and their 2-cyclotomic cosets. Once these cosets are added to $\mathcal{S}(\mathcal{C})$, it will also contain the following locality trains: $T_3 = \{6, 27, 48\}$ and $T_4 = \{12, 33, 54\}$. Note that T_2, T_3 and T_4 intersect $\mathcal{S}(\mathcal{C})$ on the subsets $\{3, 24\}$, $\{6, 27\}$ and $\{12, 33\}$, respectively. Let m_x be the number of locality trains that correspond to x and have nonempty intersection with $\mathcal{S}(\mathcal{C})$. In this example, T_1 is disjoint from $\mathcal{S}(\mathcal{C})$. Hence, $m_{21} = 3$.

In the following theorem we quantify our idea of reducing complexity by appending locality trains to the zeros of the code.

Proposition IV.1. Let \mathcal{C} be a cyclic code of composite length and let $\tau = \{x_1, x_2, \dots, x_L\}$ be a chain. Let $\mathcal{C}_{\text{new}} \subset \mathcal{C}$ be a subcode obtained by adding to $\mathcal{S}(\mathcal{C})$ the locality trains corresponding to $x \in \tau$. Suppose that the coordinates of \mathcal{C} (and \mathcal{C}_{new}) are permuted according to the ordering γ . Then, we have:

$$s_x(\mathcal{C}) - s_x(\mathcal{C}_{\text{new}}) = s_{n-x}(\mathcal{C}) - s_{n-x}(\mathcal{C}_{\text{new}}) = m_x, \quad (5)$$

where $s_i(\cdot)$ is the state complexity at i^{th} level for the code and m_x is the number of locality trains added corresponding to x , which intersect with $\mathcal{S}(\mathcal{C})$.

Proof. Fix an $x \in \tau$. Consider the code $\mathcal{C}_{\text{new}} \subset \mathcal{C}$ obtained by adding some locality trains that corresponding to x . From (3) we have $s_x(\mathcal{C}) - s_x(\mathcal{C}_{\text{new}}) = [p^x(\mathcal{C}) - p_x(\mathcal{C})] - [p^x(\mathcal{C}_{\text{new}}) - p_x(\mathcal{C}_{\text{new}})]$. By definition of γ the first x (and similarly, the last x) coordinates of the permuted codewords correspond to length- x support sets. This implies that after the permutation, the code obtained by puncturing \mathcal{C} to the first (resp., the last) x coordinates is equivalent to a punctured subcode of \mathcal{C} of length x . A similar statement holds for shortened codes as well. It follows that $p^x(\mathcal{C})$ and $p^x(\mathcal{C}_{\text{new}})$ are dimensions of length- x punctured codes and similarly, $p_x(\mathcal{C})$ and $p_x(\mathcal{C}_{\text{new}})$ are the dimensions of length- x shortened codes. The zero sets for length- x punctured and shortened codes are given in Lemma-II.2 and Lemma-II.1. As noted above, the union of locality trains that correspond to x

(together with their cyclotomic cosets) results in a disjoint union of locality trains. In other words, every zero in $\mathcal{S}(\mathcal{C}_{\text{new}}) \setminus \mathcal{S}(\mathcal{C})$ is an element in some locality train that corresponds to x .

Every locality train that corresponds to x and is disjoint from $\mathcal{S}(\mathcal{C})$, contributes a new zero for the length- x shortened and punctured codes of \mathcal{C}_{new} that is not a zero of the corresponding length- x shortened and punctured codes of \mathcal{C} . This follows from Lemma-II.2 and Lemma-II.1. It is clear that length- x punctured (and similarly, shortened) codes of \mathcal{C}_{new} are subcodes of length- x punctured (and similarly, shortened) codes of \mathcal{C} . As state complexity at level x is the difference of the dimensions of length- x shortened and punctured code, the locality trains that do not intersect $\mathcal{S}(\mathcal{C})$, will not contribute towards the difference in the state complexity at level x (respectively, $(n - x)$).

On the other hand, every added locality train corresponding to x that intersects $\mathcal{S}(\mathcal{C})$, implies the presence of a new zero for length- x punctured code of \mathcal{C}_{new} , whereas for the length- x shortened code of \mathcal{C}_{new} it does not make any difference. This again follows from Lemma-II.2 and Lemma-II.1. Hence, every locality train corresponding to x that intersect with $\mathcal{S}(\mathcal{C})$ results in the increase of the difference in the state complexity at level x (or $(n - x)$) by 1, and the proof follows. \square

Next we derive a bound on the state complexity of cyclic codes. We begin with explaining the idea. Let \mathcal{C} be a cyclic code of composite length (with or without locality). We estimate the state complexity of the permuted code (suppressing the difference between the two versions of the code before and after the permutation). Suppose that the coordinates of the code \mathcal{C} are permuted with respect to some chain $\tau = \{x_1, \dots, x_L\}$. It is possible to bound above the state complexity s_i relying on its set of zeros $\mathcal{S}(\mathcal{C})$. Indeed, consider the sections of the trellis at the indices in the set τ . The corresponding values of the state complexity are governed by the dimensions of the respective past and future codes. From Lemmas II.1 and -II.2, the dimensions of the past codes $\{p^{x_1}, p^{x_2}, \dots, p^{x_L}\}$ and $\{p_{x_1}, p_{x_2}, \dots, p_{x_L}\}$ can be obtained. Past punctured code dimensions can be expressed in terms of future shortened code dimensions using the rank-nullity theorem, namely

$$p^{n-j} = k - f_{n-j}, \quad j \in [0, n].$$

Furthermore, $f_{n-x} = p_x$ and $f^{n-x} = p^x$, for $x \in \tau$. This follows because the last x coordinates form a length- x support set. Hence the code obtained by puncturing (and similarly, shortening) to the last x coordinates is permutation-equivalent to length- x punctured (resp., shortened) codes of \mathcal{C} . Therefore, we can also find the dimensions of the shortened codes $\{p^{n-x_1}, p^{n-x_2}, \dots, p^{n-x_L}\}$.

It is straightforward to see that $\{p^i : i \in [0, n]\}$ is a non-decreasing sequence bounded above by k , and the maximum value of the increase within the x -support set is p^x . Also, $p^0 \triangleq 0$ and $p^{i+1} - p^i \in \{0, 1\}$. This enables us to construct a sequence $\{\phi_i\}_{i=0}^n$ which forms an upper bound for $\{p^i\}_{i=0}^n$. We begin with defining ϕ_i for the known instances of i :

$$\phi_i = p^i, \quad i \in \{0, x_1, \dots, x_L, n - x_L, \dots, n - x_1, n\}$$

To fill in the gaps in the sequence, $\{\phi_i\}$ we use the constraints on $\{p^i\}$ mentioned above. Namely, starting with $i = 0$, we either put $\phi_{i+1} = \phi_i$ or $\phi_{i+1} = \phi_i + 1$, as appropriate. Clearly we have

$$p^i \leq \phi_i \text{ and } f^i \leq \phi_{n-i}$$

(the second inequality implied by symmetry). Now let $\mu_i = \phi_i + \phi_{n-i} - k$ and observe from (3) that $s_i \leq \mu_i$.

Similar relations can be obtained for the dual code using the well-known relation between $\mathcal{S}(\mathcal{C})$ and $\mathcal{S}(\mathcal{C}^\perp)$ ($\beta \in \mathcal{S}(\mathcal{C})$ iff $\beta^{-1} \notin \mathcal{S}(\mathcal{C}^\perp)$). Using this together with the fact that the state complexity of \mathcal{C} and \mathcal{C}^\perp is the same [25], we obtain the bound

$$s_i \leq \min\{\mu_i, \mu_i^\perp\}$$

with μ_i^\perp defined accordingly. Note that this bound is tight for $i \in \{0, x_1, x_2, \dots, x_L, n - x_L, n - x_{L-1}, \dots, n - x_1, n\}$.

We can make this argument more explicit by stating an upper bound on the maximum state complexity of the code.

Theorem IV.2. *Let \mathcal{C} be an $[n, k]$ cyclic code with composite length n . Assume that the coordinates of \mathcal{C} are γ -ordered based on a chain $\tau = \{x_1, x_2, \dots, x_L\}$. Then*

$$\max_{i \in [0, n]} s_i \leq \min \left\{ k, n - k, \sum_{i=1}^L \left(\frac{x_{i+1}}{x_i} - 1 \right) p^{x_i} + x_1 - k, \sum_{i=1}^L \left(\frac{x_{i+1}}{x_i} - 1 \right) (x_i - p_{x_i}) + x_1 - (n - k) \right\}$$

where we put $x_{L+1} = n$ by definition.

The proof of this theorem is given in Appendix-B.

The third term under the $\min\{\}$ can be rewritten as $\sum_{i=0}^L h_i$, where $h_i \triangleq \frac{x_{i+1}}{x_i} p^{x_i} - p^{x_{i+1}}$, $i > 0$, $h_0 \triangleq x_1 - p^{x_1}$ and $p^{x_{L+1}} = k$. In other words, $h_i, i > 0$ indicates the global parities for the length- x_{i+1} punctured code with respect to smaller length- x_i punctured codes. A simple calculation yields that $n - k = h_0 \frac{n}{x_1} + h_1 \frac{n}{x_2} + h_2 \frac{n}{x_3} + \dots + h_L$. This suggests that, for a given n and k , the sum $\sum_{i=0}^L h_i$ can be minimized by making h_i 's larger for smaller i 's (i.e., more locality). The fourth expression accounts for a similar argument about locality of the dual code.

0					
1	2	4	8	16	32
3	6	12	24	48	33
5	10	20	40	17	34
7	14	28	56	49	35
9	18	36			
11	22	44	25	50	37
13	26	52	41	19	38
15	30	60	57	51	39
21	42				
23	46	29	58	53	43
27	54	45			
31	62	61	59	55	47

Table I: 2-cyclotomic cosets modulo 63

(n, k)	No. of edges	No. of vertices	Approx. computations per information bit (real additions+comparisons)	s_{21}, s_{42}	$\max_i s_i$
(63, 51)	122876	65534	2409 + 1124	9	12
(63, 48)	29180	15998	608 + 275	6	9

Table II: Trellis complexities of codes \mathcal{C} and \mathcal{C}_1 in Example IV.2 with respect to $\tau = \{3, 21\}$

Example IV.2. Let \mathcal{C} be the $[63, 51]$ binary BCH code with defining zeros at $\{1, 3\}$ and their 2-cyclotomic cosets (see Table-I). We add three locality trains $\mathcal{T}_3^{(21)}$, $\mathcal{T}_6^{(21)}$ and $\mathcal{T}_{12}^{(21)}$ to $\mathcal{S}(\mathcal{C})$, to obtain the $[63, 48]$ BCH-like code \mathcal{C}_1 . All these trains intersect $\mathcal{S}(\mathcal{C})$ and hence, from Proposition-IV.1, $s_{21}(\mathcal{C}) - s_{21}(\mathcal{C}_1) = 3$. In Table II and Fig. 6, we consider the γ -ordering based on $\tau = \{3, 21\}$ (which gives the least computational complexity for \mathcal{C} and \mathcal{C}_1 among all possible τ 's). In order to stress the need for careful locality addition, we consider adding \mathbb{C}_7^{63} to \mathcal{C} to obtain the $(63, 45)$ BCH-like code \mathcal{C}_2 . There are two locality trains $\mathcal{T}_7^{(21)}$ and $\mathcal{T}_{14}^{(21)}$ introduced here, with none of them intersecting $\mathcal{S}(\mathcal{C})$. In this case, it turns out that, even under the best chain-based permutation for \mathcal{C}_2 , approximately 12265 computations (8465 real additions + 3800 comparisons) per information bit are required to decode \mathcal{C}_2 . For \mathcal{C} , \mathcal{C}_1 and \mathcal{C}_2 , Theorem IV.2 accurately predicts the maximum state complexity.

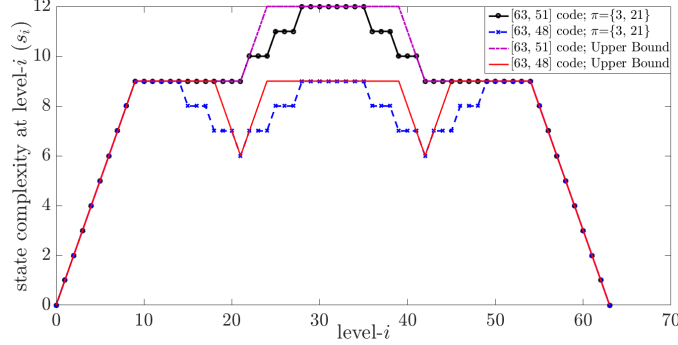


Fig. 6: State complexity profile and upper bound for codes \mathcal{C} and \mathcal{C}_1 of Example IV.2.

APPENDIX

A. Proof of Lemma-II.2

Let $\underline{c} = (c_0, c_1, \dots, c_{n-1}) \in \mathcal{C}$ be a codeword and let $c(x) = \sum_{i=0}^{n-1} c_i x^i$ be the corresponding polynomial. Recall the relation between \underline{c} and its frequency domain representation $\hat{\underline{c}} = (\hat{c}_0, \hat{c}_1, \dots, \hat{c}_{n-1})$ [27, Ch.6]: For $\lambda = 0, 1, \dots, n-1$

$$\hat{c}_\lambda = c(\alpha^\lambda) = \sum_{t=0}^{n-1} c_t \alpha^{t\lambda} \quad \text{and} \quad c_\lambda = \frac{1}{n} \sum_{j=0}^{n-1} \hat{c}_j \alpha^{-\lambda j}.$$

Therefore $\lambda \in \mathcal{S}(\mathcal{C})$ iff $\hat{c}_\lambda = 0$ for all $\underline{c} \in \mathcal{C}$.

Now let $\underline{a} = (a_0, a_1, \dots, a_{n_1-1})$ be the codeword obtained by puncturing \underline{c} to the n_1 coordinates $\{0, \nu, 2\nu, \dots, n - \nu\}$. Let $\beta = \alpha^\nu$ be a primitive n_1^{th} root of unity. The frequency domain representation of \underline{a} is given by:

$$\begin{aligned}
\hat{a}_\lambda &= \sum_{t=0}^{n_1-1} a_t \beta^{\lambda t} \\
&= \sum_{t=0}^{n_1-1} c_{\nu t} \alpha^{\nu \lambda t} \\
&= \frac{1}{n} \sum_{t=0}^{n_1-1} \sum_{j=0}^{n-1} \hat{c}_j \alpha^{-\nu j t} \alpha^{\nu \lambda t} \\
&= \frac{1}{n} \sum_{j=0}^{n-1} \hat{c}_j \sum_{t=0}^{n_1-1} \alpha^{-\nu t(j-\lambda)} \\
&= \frac{n_1}{n} \sum_{j=0}^{n-1} \hat{c}_j \left[\mathbb{1}(j = \lambda \pmod{n_1}) \right], \quad \lambda = 0, 1, \dots, n_1 - 1.
\end{aligned} \tag{6}$$

Thus $\lambda \pmod{n_1} \in \mathcal{S}(\mathcal{C}_{\mathcal{I}_i})$ if $\lambda \in \mathcal{S}(\mathcal{C})$, i.e., the elements of $\mathcal{S}(\mathcal{C})$ are of the form $j + l n_1, l \in [0, \nu]$.

Now let $\lambda = \tilde{\lambda} + l n_1$, where $l \in [0, \nu - 1]$ and $\tilde{\lambda} \in \mathcal{S}(\mathcal{C}_{\mathcal{I}_i})$. For $\lambda = 0, 1, \dots, n_1 - 1$ we have the following:

$$\begin{aligned}
\hat{c}_\lambda &= \sum_{t=0}^{n-1} c_t \alpha^{\lambda t} \\
&= \sum_{i=0}^{\nu-1} \sum_{j=0}^{n_1-1} c_{i+j\nu} \alpha^{\lambda(i+j\nu)} \\
&= \sum_{i=0}^{\nu-1} \alpha^{i\lambda} \sum_{j=0}^{n_1-1} c_{i+j\nu} \beta^{\lambda j} \\
&= \sum_{i=0}^{\nu-1} \alpha^{i(\tilde{\lambda} + l n_1)} \left[\sum_{j=0}^{n_1-1} c_{i+j\nu} \beta^{\tilde{\lambda} j} \right] \\
&= 0
\end{aligned} \tag{7}$$

Taken together, (6) and (7) imply our claim.

B. Proof of Theorem-IV.2

The first two terms follow from (4). The third term is obtained by a counting argument involving the total number of length- x_l punctured codes of dimensions $\{p^{x_l}\}$, $l \in [L]$, supplied by $\mathcal{P}^{(i)}$ and $\mathcal{F}^{(i)}$ at any level- i . Let \mathcal{C}_{x_l} denote the length- x_l punctured code of dimension p^{x_l} . We first note the following fact: $\frac{x_b}{x_a} p^{x_a} \geq p^{x_b}$, $\forall x_a, x_b : x_a \leq x_b$. This is true because, for each $\lambda \in \mathcal{S}(\mathcal{C}_{x_a})$, $\{\lambda, \lambda + x_a, \lambda + 2x_a, \dots, \lambda + x_b - x_a\} \in \mathcal{S}(\mathcal{C}_{x_b})$, from Lemma-II.2. Hence, $p^{x_b} = x_b - |\mathcal{S}(\mathcal{C}_{x_b})| \leq x_b - \frac{x_b}{x_a} |\mathcal{S}(\mathcal{C}_{x_a})| = \frac{x_b}{x_a} (x_a - |\mathcal{S}(\mathcal{C}_{x_a})|) = \frac{x_b}{x_a} p^{x_a}$. If $x_l \nmid i$ for all $l \in [L]$, after counting the smaller punctured codes that contribute to p^i and f^i , it can be seen that $s_i \leq \sum_{i=1}^L \left(\frac{x_{i+1}}{x_i} - 1 \right) p^{x_i} + x_1 - k$. Now suppose that $x_L \mid i$. We have the following chain of inequalities:

$$\begin{aligned}
s_i &\leq p^i + f^i - k \\
&\leq \frac{n}{x_L} p^{x_L} - k \\
&= \left(\frac{n}{x_L} - 1 \right) p^{x_L} + p^{x_L} - k \\
&\leq \left(\frac{n}{x_L} - 1 \right) p^{x_L} + \left(\frac{x_L}{x_{L-1}} \right) p^{x_{L-1}} - k \\
&= \left(\frac{n}{x_L} - 1 \right) p^{x_L} + \left(\frac{x_L}{x_{L-1}} - 1 \right) p^{x_{L-1}} + p^{x_{L-1}} - k \\
&\leq \left(\frac{n}{x_L} - 1 \right) p^{x_L} + \left(\frac{x_L}{x_{L-1}} - 1 \right) p^{x_{L-1}} + \dots + \left(\frac{x_2}{x_1} \right) p^{x_1} - k \\
&\leq \sum_{i=1}^L \left(\frac{x_{i+1}}{x_i} - 1 \right) p^{x_i} + x_1 - k
\end{aligned}$$

If $x_l \nmid i \forall l \in \{a, a+1, \dots, L\}$ and $x_{a-1} \mid i$,

$$\begin{aligned}
s_i &\leq p^i + f^i - k \\
&\leq \left(\frac{n}{x_L} - 1\right)p^{x_L} + \left(\frac{x_L}{x_{L-1}} - 1\right)p^{x_{L-1}} + \dots + \left(\frac{x_a}{x_{a-1}} - 1\right)p^{x_{a-1}} - k \\
&= \left(\frac{n}{x_L} - 1\right)p^{x_L} + \left(\frac{x_L}{x_{L-1}} - 1\right)p^{x_{L-1}} + \dots + \left(\frac{x_a}{x_{a-1}} - 1\right)p^{x_{a-1}} + p^{x_{a-1}} - k \\
&\leq \left(\frac{n}{x_L} - 1\right)p^{x_L} + \dots + \left(\frac{x_a}{x_{a-1}} - 1\right)p^{x_{a-1}} + \left(\frac{x_{a-1}}{x_{a-2}} - 1\right)p^{x_{a-2}} - k \\
&\leq \sum_{i=1}^L \left(\frac{x_{i+1}}{x_i} - 1\right)p^{x_i} + x_1 - k
\end{aligned}$$

As $x_i \mid x_j$ for all i, j such that $1 \leq i \leq j \leq L$, these conditions essentially cover all the cases and the proof follows. The fourth term is obtained by repeating the same analysis for the dual code.

REFERENCES

- [1] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate (Corresp.)," *IEEE Transactions on Information Theory*, vol. 20, no. 2, pp. 284–287, 1974.
- [2] J. K. Wolf, "Efficient maximum likelihood decoding of linear block codes using a trellis," *IEEE Transactions on Information Theory*, vol. 24, no. 1, pp. 76–80, 1978.
- [3] D. Chase, "A class of algorithms for decoding block codes with channel measurement information," *IEEE Transactions on Information Theory*, vol. 18, no. 1, pp. 170–182, 1972.
- [4] M. P. C. Fossorier and S. Lin, "Soft-decision decoding of linear block codes based on ordered statistics," *IEEE Transactions on Information Theory*, vol. 41, no. 5, pp. 1379–1396, 1995.
- [5] S. Lin and D. J. Costello, *Error Control Coding*. Pearson Prentice Hall, 2004.
- [6] J. Hagenauer, E. Offer, and L. Papke, "Iterative decoding of binary block and convolutional codes," *IEEE Transactions on Information Theory*, vol. 42, no. 2, pp. 429–445, 1996.
- [7] P. Gopalan, C. Huang, H. Simitci, and S. Yekhanin, "On the Locality of Codeword Symbols," *IEEE Transactions on Information Theory*, vol. 58, no. 11, pp. 6925–6934, 2012.
- [8] A. Agarwal, A. Barg, S. Hu, A. Mazumdar, and I. Tamo, "Combinatorial alphabet-dependent bounds for locally recoverable codes," preprint, 2017, arXiv:1702.02685.
- [9] I. Tamo, A. Barg, and A. Frolov, "Bounds on the parameters of locally recoverable codes," *IEEE Transactions on Information Theory*, vol. 62, no. 6, pp. 3070–3083, 2016.
- [10] M. P. C. Fossorier, "Iterative reliability-based decoding of low-density parity check codes," *IEEE Journal on Selected Areas in Communications*, vol. 19, no. 5, pp. 908–917, 2001.
- [11] A. Kothiyal, O. Y. Takeshita, W. Jin, and M. P. C. Fossorier, "Iterative reliability-based decoding of linear block codes with adaptive belief propagation," *IEEE Communications Letters*, vol. 9, no. 12, pp. 1067–1069, 2005.
- [12] J. Jiang and K. R. Narayanan, "Iterative Soft-Input Soft-Output Decoding of Reed-Solomon Codes by Adapting the Parity-Check Matrix," *IEEE Transactions on Information Theory*, vol. 52, no. 8, pp. 3746–3756, 2006.
- [13] G. M. Kamath, N. Prakash, V. Lalitha, and P. V. Kumar, "Codes with local regeneration and erasure correction," *IEEE Transactions on Information Theory*, vol. 60, no. 8, pp. 4637–4660, 2014.
- [14] B. Sasidharan, G. K. Agarwal, and P. V. Kumar, "Codes with hierarchical locality," in *IEEE International Symposium on Information Theory*, 2015, pp. 1257–1261.
- [15] A. Wang and Z. Zhang, "Repair Locality With Multiple Erasure Tolerance," *IEEE Transactions on Information Theory*, vol. 60, no. 11, pp. 6979–6987, 2014.
- [16] J. L. Massey, *Threshold Decoding*, 1963.
- [17] I. Tamo, A. Barg, S. Goparaju, and R. Calderbank, "Cyclic LRC codes, binary LRC codes, and upper bounds on the distance of cyclic codes," *International Journal of Information and Coding Theory*, vol. 3, no. 4, pp. 345–364, 2016.
- [18] I. Tamo and A. Barg, "A family of optimal locally recoverable codes," *IEEE Transactions on Information Theory*, vol. 60, no. 8, pp. 4661–4676, 2014.
- [19] A. Vardy and Y. Be'ery, "Maximum-likelihood soft decision decoding of BCH codes," *IEEE Transactions on Information Theory*, vol. 40, no. 2, pp. 546–554, 1994.
- [20] B. Chen, S.-T. Xia, J. Hao, and F.-W. Fu, "Constructions of Optimal Cyclic (r, δ) Locally Repairable Codes," preprint, 2016, arXiv:1609.01136.
- [21] M. Lentmaier and K. S. Zigangirov, "On generalized low-density parity-check codes based on Hamming component codes," *IEEE Communications Letters*, vol. 3, no. 8, pp. 248–250, 1999.
- [22] J. Boutros, O. Pothier, and G. Zémor, "Generalized low density (Tanner) codes," in *IEEE International Conference on Communications*, vol. 1, 1999, pp. 441–445.
- [23] A. Barg, I. Tamo, and S. Vlăduț, "Locally recoverable codes on algebraic curves," preprint, 2016, arXiv:1603.08876. A 5-page extended abstract is available in Proceedings of the IEEE International Symposium on Information Theory, 2015, pp. 1252–1256.
- [24] R. J. McEliece, "On the BCJR trellis for linear block codes," *IEEE Transactions on Information Theory*, vol. 42, no. 4, pp. 1072–1092, 1996.
- [25] G. D. Forney Jr., "Coset codes-II: Binary lattices and related codes," *IEEE Transactions on Information Theory*, vol. 34, no. 5, pp. 1152–1187, 1988.
- [26] D. J. Muder, "Minimal trellises for block codes," *IEEE Transactions on Information Theory*, vol. 34, no. 5, pp. 1049–1053, 1988.
- [27] R. E. Blahut, *Algebraic codes for data transmission*. Cambridge University Press, 2003.